REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
01-01-2012	Final	09/01/2009 - 01/12/2012

4. TITLE AND SUBTITLE

Acquiring and Exploiting Rich Casual Modes for Robust Decision Making

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-09-1-0627

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Tenenbaum, Joshua B.
Kaelbling, Leslie P.
Littman, Michael L.
Wingate, David

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

MIT, Cambridge, MA
Rutgers University, Piscataway, NJ

8. PERFORMING ORGANIZATION REPORT NUMBER

(none)

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Office of Scientific Research
875 North Randolph Street
Suite 325, Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

AFRL-OSR-VA-TR-2012-0735

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution A - Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Our project has made fundamental contributions to the understanding of robust decision making in human beings and machines through an intensive examination of how to learn rich, causal models of the world and how agents can use those models to make decisions. We report progress in eight key areas: 1) Significant progress on building rich models using probabilistic programming. 2) New Bayesian nonparametric models for learning dynamical systems. 3) A new Bayesian model that learns optimal policies by combining expert demonstrations of optimal behavior and data gathered by non-optimal exploration. 4) New policy learning methods based on probabilistic search. 5) A new policy learning algorithm for Bayesian reinforcement learning which is provably efficient. 6) New algorithms for hierarchical planning. 7) New transfer learning models which use hierarchical knowledge totransfer abstract properties of domains, such as general notions of consistency, determinism, generalizability, or clusterability. 8) A new foundation for compositional transfer of policy fragments.

15. SUBJECT TERMS

Hierarchical Bayesian models, probabilistic programming, reinforcement learning, transfer learning, policy priors, hierarchical planning.

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*

Acquiring and Exploiting Rich Causal Models for Robust Decision Making: Final Report

Leslie Pack Kaelbling Joshua B. Tenenbaum Michael L. Littman

1 Summary of Contributions

Our project has made fundamental contributions to the understanding of robust decision making in human beings and machines through an intensive examination of how to learn rich, causal models of the world and how agents can use those models to make decisions. Our core assumption is that both humans and machines can be viewed as boundedly rational agents who are attempting to maximize their expected utility based on their current state of knowledge subject to their own computational limitations. This has motivated our key technical approach: we have focused on **probabilistic models** to explicitly deal with uncertainty; **hierarchical algorithms** to leverage multiple levels of abstraction; and **factored models** to capture rich structure in complex problems. We have additionally made significant contributions to **probabilistic programming** languages that allow agent designers to quickly specify complex probabilistic models.

Our research is spanning the entire spectrum of learning and decision making, from low-level dynamical systems modeling to state estimation and optimal control. We are pleased to report progress in eight key areas:

- Significant progress on building rich models using probabilistic programming. We have developed new implementation strategies [1], new inference methods [2], and made deep connections between problem formulations [3].
- Bayesian nonparametric models for learning dynamical systems. We have contributed four new models of dynamical systems: two that use state clustering to quickly estimate parameters of unknown dynamical models [4], and two that learn factored causal models [5, 6].
- A Bayesian model that learns optimal policies by combining expert demonstrations of optimal behavior and data gathered by non-optimal exploration [2]. We use nonparametrics to model the expert's policy, and fuse that policy information with a model of world dynamics, and show that performance improves with the addition of expert examples.
- Policy learning methods based on probabilistic search [7]. We have contributed new ideas about how to search structured policy spaces to help solve complex control problems, such as those faced by articulated agents in complicated environments.

- A policy learning algorithm for Bayesian reinforcement learning which is provably efficient [4]. Our new BOSS algorithm optimistically mixes sampled models together to construct a policy which is guaranteed to result in either high reward or more information about an unknown system, which can be used to further improve performance.
- Hierarchical planning. We have contributed a new algorithm which performs optimistic top-down planning while only partially committing to specific sequences of low-level actions [8]. This model connects high-level, relational properties of domains with low-level geometric knowledge, resulting in fast planning in complex, richly structured problems.
- Transfer learning models which use hierarchical knowledge to transfer abstract properties of domains, such as general notions of consistency, determinism, generalizability, or clusterability. Experience gained in training worlds can be transferred to test worlds, resulting in accelerated learning and improved control performance. Additional experiments with human subjects suggest that the model makes qualitatively human-like decisions.
- A foundation for compositional transfer of policy fragments. We have shown how natural language grammatical formalisms can extract structure at multiple levels of abstractions in families of domains considered by traditional reinforcement learning formalisms [9].

The following sections provide detail about each of the eight main progress areas.

2 Bayesian Models of Dynamical Systems

An important part of robust decision making is understanding the consequences of decisions. This requires a model of the world, and when such a model is not given, it must be learnt from experience. Here, we discuss how to deal with uncertainty and noise in the model building process, while incorporating prior knowledge.

2.1 The Infinite Latent Events Model

We have developed the Infinite Latent Events Model [5], a nonparametric hierarchical Bayesian distribution over infinite dimensional Dynamic Bayesian Networks, which can capture rich, factored structure in temporal data. These DBNs have with binary state representations and noisy-OR-like transitions. The model operates on "events," which are discrete factors that work together both to create new events and to generate observations.

The distribution can be used to learn structure in discrete timeseries data by simultaneously inferring the set of latent events, which events fired at each timestep, how those events are causally linked, and how the events combine to form observations. Examples of the model in action are shown in Fig. 1. On the left is a stylized representation of the goal of the model: given the sequence of images, the model learns that there are four latent events (represented by stylized glyphs), and learns causal and observation structure. The right shows an example factorization of images.

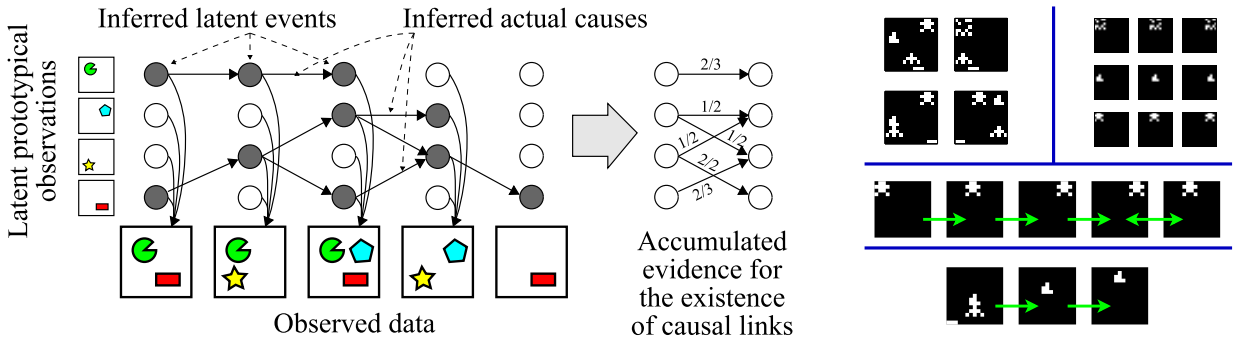

Figure 1: On the left: the ILEM can learn timeseries structure. Given observations (represented here as images) it infers latent events, actual causes, and prototypical latent observations. On the right: results on a simple video sequence akin to "space invaders." Top left: typical observations. Top right: several of the prototypical observations associated with each event: the top row shows explosions, the middle row shows some bullet events, and the bottom row shows the alien. Bottom figure: two chains of causal events inferred by the model. The alien moves back and forth, and a bullet moves upward after being fired from the ship.

The notion of event is deliberately abstract, meaning the model can be applied in many domains. We illustrate the model on a sound factorization task, a network topology identification task, and a video game task. In each domain, the model correctly discovers the latent "events": in the sound domain, latent events correspond to unmixed sound signals; in the video game task, they correspond to sprites, and in the network topology task, they correspond to computers. We have also applied the ILEM to neural spike train data, in which the events correspond to clusters of neurons firing.

2.2 Infinite Dynamical Bayesian Networks

While the ILEM is conceptually and practically powerful, it still has limitations. Even the ILEM's factorization of the world is not sufficient, for example, to capture structure in the real world such as objects. Our next model, the *Infinite Dynamic Bayesian Network* [6], incorporates a more flexible factorization and richer observation spaces by learning dynamic Bayes nets (DBNs) of arbitrary cardinality.

We use Bayesian nonparametrics to learn every aspect of a classic DBN:

- How many latent factors should there be?
- How many different values can each factor take on?
- What conditional independencies exist between factors?
- What is the transition distribution between factors?

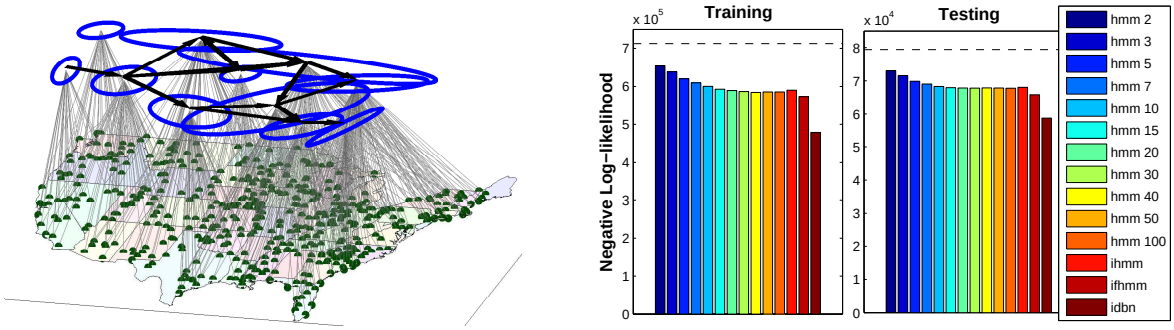

Figure 2: On the left: an example of the factorization learned by the iDBN. Blue circles represent latent factors; grey arrows represent connections between these factors and the observation dimensions (the weather stations). Thick black arrows represent the causal connections between the latent factors. The strong west-to-east causality is the result of the North American jet stream.

- How do factors combine to form observations?

Inference in this model is challenging. We use a combination of Gibbs sampling, loopy belief propagation, forward-filtering with backward sampling, and basic MH-based MCMC.

We have applied this model to a variety of datasets, including songbird data, weather data, and a variety of synthetic datasets. Figure 2 illustrates the kind of models that can be learned: given an input timeseries representing precipitation levels of different weather stations, the iDBN successfully recovers a reasonable representation of the causual structure of weather patterns in the US (note that no geographical information was given to the model). It also performs much better than a variety of other timeseries models.

3 Beyond Graphical Models: Advances in Probabilistic Programming

Probabilistic models provide a rich and principled framework for learning and inference. But actually coding up inference algorithms is tedious and error-prone. Probabilistic programming is a declarative formalism for specifying stochastic, generative processes. Probabilistic programming languages allow modelers to specify a stochastic process using syntax that resembles modern programming languages. This then allows a "probability compiler" to do the heavy lifting of crafting inference algorithms. Our research has pushed every aspect of these languages.

3.1 Implementing Probabilistic Programming Languages

Our first contribution [1] is a general method of transforming arbitrary programming languages into probabilistic programming languages with straightforward MCMC inference engines. Ran-

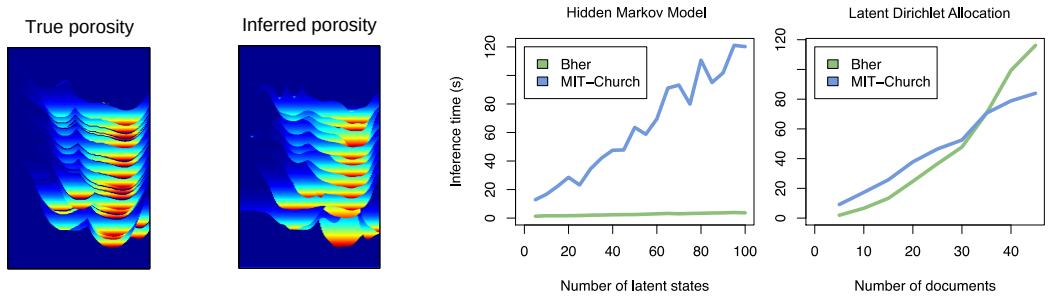

Figure 3: On the left: results on modeling rock porosity. On the right: Inference performance for the HMM (10,000 samples) and LDA model (1,000 samples).

dom choices in the program are "named" with information about their position in an execution trace; these names are used in conjunction with a database holding values of random variables to implement MCMC inference in the space of execution traces. We encode naming information using lightweight source-to-source compilers. Our method enables us to reuse existing infrastructure (compilers, profilers, etc.) with minimal additional code, implying fast models with low development overhead.

The method is simple and fast, and the resulting languages permit a great deal of flexibility in the specification of distributions by mixing stochastic and deterministic elements with arbitrary language features (such as objects, inheritance, operator overloading, closures, recursion, libraries, etc.). Our example implementations have compact code bases and reasonable inference speed.

We illustrated the technique on two languages, one functional and one imperative: Bher, a compiled version of the Church language which eliminates interpretive overhead of the original MIT-Church implementation, and Stochastic Matlab, a new open-source language.

Empirically, we have applied the resulting inference engines to a variety of problems. Figure 3 shows some results where we have applied our technique to geophysical modeling, topic modeling and timeseries modeling.

The main directions for improvement are better mixing and faster inference. Because source-to-source transformations are often compositional, more transformations could be applied to enhance performance. These could reduce redundant computation between traces, generate compound proposals, or implement constraint propagation for initializing conditioners. Future work will investigate these issues, as well as the possibility of new languages, such as stochastic Python.

3.2 Efficient Inference: Nonstandard Interpretations

Scalable inference is the primary challenge in probabilistic programming. In probabilistic modeling more generally, inference is efficient when we can take advantage of structure in a distribution. But how can we find and exploit structure in the distribution represented by a probabilistic program?

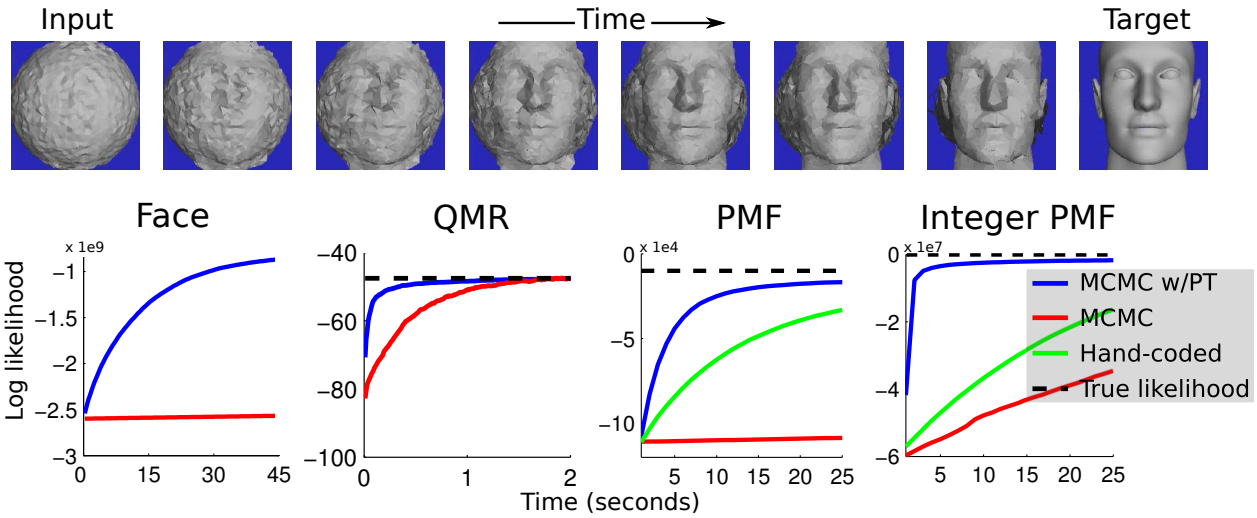

Figure 4: Top: Frames from the face task. Bottom: results on Face, QMR, PMF and Integer PMF.

Because probabilistic programs are in machine-readable format, a variety of techniques from compiler design and program analysis can be used to examine the structure of the distribution represented by the probabilistic program. We have shown how *nonstandard interpretations* of probabilistic programs can be used to craft efficient inference algorithms: information about the structure of a distribution (such as gradients or dependencies) is generated as a monad-like side computation while executing the program [2]. These interpretations can be easily coded using special-purpose objects and operator overloading. We implemented two examples of nonstandard interpretations in two different languages, and use them as building blocks to construct inference algorithms: automatic differentiation, which enables gradient based methods, and provenance tracking, which enables efficient construction of global proposals.

Empirically, we have implemented two such interpretations and demonstrated how this information can be used to find regions of high likelihood quickly, and how it can be used to generate samples with improved statistical properties versus random-walk style MCMC.

Figure 4 shows some results using a new provenance tracking technique in conjunction with a factored multiple-try MH algorithm. This method allows MCMC algorithms to construct efficient global proposals for discrete or continuous variables, and is competitive with hand-coded samplers. The figure illustrates one simple application, where a mesh is stochastically deformed until it renders to an image that looks like a target image.

More generally, this work begins to illuminate the close connections between probabilistic inference and programming language theory. It is likely that other techniques from compiler design and program analysis could be fruitfully applied to inference problems in probabilistic programs.

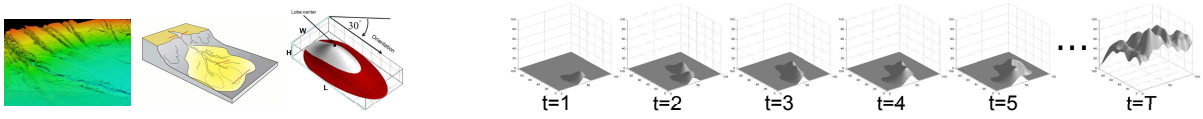

Figure 5: Example realization of the geology model. On the left: an underwater sedimentary shelf is created as rivers deposit sediments. Each deposit is approximated by a lobe. On the right: a sequence of lobes builds up over time, generating a rock volume.

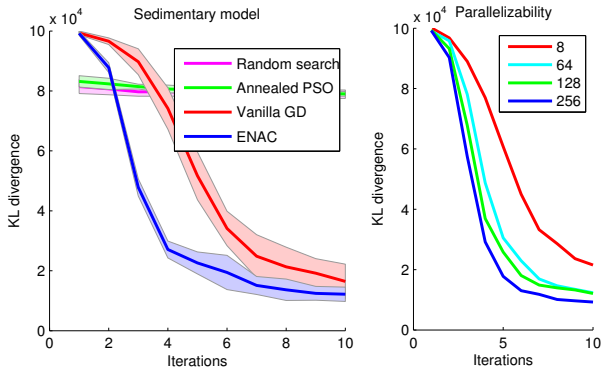

Figure 6: (Left) Results on the sedimentary model, comparing four different optimization algorithms. The RL-based ENAC algorithm does the best job. (Right) Parallelization results on the ENAC algorithm. Performance improves as more cores are added.

3.3 Efficient Inference: Automatic Variational Optimization

Basic MCMC is the state-of-the-art inference method for many probabilistic programming languages because of its simplicity, compositionality and universality. But MCMC is not a particularly efficient inference algorithm in general. Can we move beyond it?

Our next contribution [3] is a control theoretic perspective on variational inference in generative models. This connects variational inference to a temporal credit assignment problem that can be solved using reinforcement learning. The method is effective for distributions which are not analytically tractable, including highly structured distributions that arise in probabilistic programs. In addition, we have shown how to automatically derive mean-field probabilistic programs and optimize them, and demonstrate that our reinforcement learning perspective improves inference efficiency over other approaches.

Figure 5 and 6 show the results on a sophisticated model from geophysics. Here, we see that the RL-based ENAC algorithm outperforms other inference methods.

Because variational inference can be viewed as an RL problem, RL algorithms bring new tools to inference problems. This is especially appropriate in the context of deep generative models with complex structure, where RL can help propagate information backwards through the process. Future work will investigate more RL algorithms and their properties when applied to variational inference.

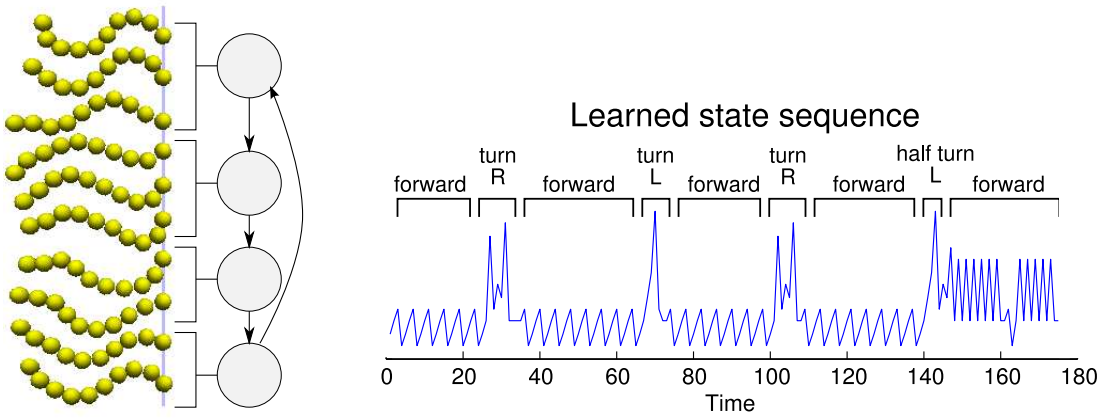

Figure 7: Left: a wiggle-like motor primitive learned by the snake agent. Right: the learned state sequence.

4 Planning Under Model Uncertainty

Given a posterior distribution over possible world models, an agent must still decide how to act. We now turn to our contributions in the area of decision making in a variety of Bayesian contexts.

4.1 Bayesian Policy Search with Policy Priors

We first considered the problem of learning to act in partially observable, continuous-state-and-action worlds where we have abstract prior knowledge about the structure of the optimal policy in the form of a distribution over policies [7]. Using ideas from planning-as-inference reductions and Bayesian unsupervised learning, we cast Markov Chain Monte Carlo as a stochastic, hill-climbing policy search algorithm. Importantly, this algorithm's search bias is directly tied to the prior and its MCMC proposal kernels, which means we can draw on the full Bayesian toolbox to express the search bias, including nonparametric priors and structured, recursive processes like grammars over action sequences.

Furthermore, we can reason about uncertainty *in the search bias itself* by constructing a hierarchical prior and reasoning about latent variables that determine the abstract structure of the policy. This yields an adaptive search algorithm—our algorithm *learns to learn* a structured policy efficiently. We showed how inference over the latent variables in these *policy priors* enables intra- and intertask transfer of abstract knowledge. We demonstrate the flexibility of this approach by learning meta search biases, by constructing a nonparametric finite state controller to model memory, by discovering motor primitives using a simple grammar over primitive actions, and by combining all three.

Figure 7 illustrates some results from this line of work. It shows the policy learned by a Bayesian model of motor primitives when applied to a snake robot in a maze-like world. The models learns a set of motion primitives that look like the serpentine motions of real snakes – because those are

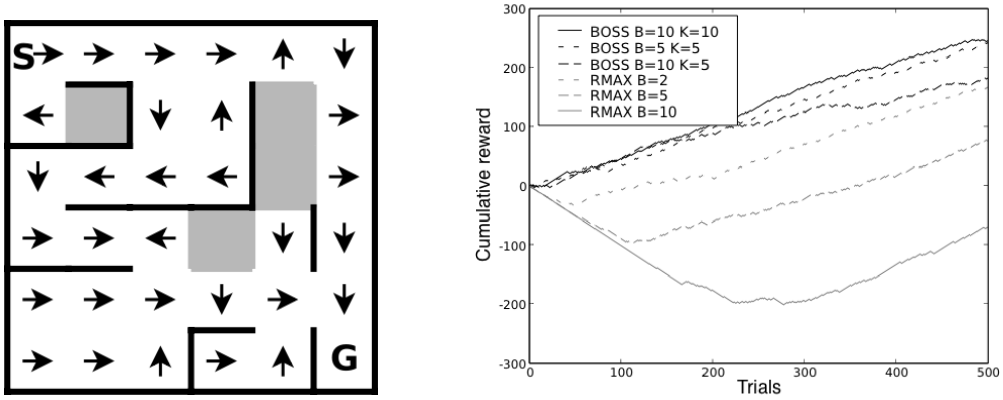

Figure 8: On the left: a simple maze world with shared structure between states. States are fully observable, but are augmented with information about the presence or absence of walls. On the right: comparison of the BOSS algorithm with RMAX (a leading competitor) for different values of a key parameter. BOSS learns more quickly under a wide variety of parameter settings.

the naturally optimal dynamics of motion for this kind of agent.

As planning problems become more complex, we believe that it will become increasingly important to be able to reliably and flexibly encode abstract prior knowledge about the form of optimal policies into search algorithms. Encoding such knowledge in a policy prior has allowed us to combine unsupervised, hierarchical Bayesian techniques with policy search algorithms. This combination accomplished three things: first, we have shown how we can express abstract knowledge about the form of a policy using nonparametric, structured, and compositional distributions (in addition, the policy prior implicitly expresses a preference ordering over policies). Second, we have shown how to incorporate this abstract prior knowledge into a policy search algorithm based on a reduction from planning to MCMC-based sampling. Third, we have shown how hierarchical priors can adaptively direct the search for policies, resulting in accelerated learning. Future work will address computational issues and push the algorithm to solve more challenging planning problems. We currently use a generic probabilistic modeling language and inference algorithm; this genericity is a virtue of our approach, but special purpose engines could accelerate learning.

4.2 BOSS: Planning in Bayesian RL

The ILEM builds a model of a dynamical system, but does not consider planning in it. Conditioned on data from the system, the ILEM yields samples from a posterior distribution over models. How can these sampled models be used for planning? We developed a modular approach to reinforcement learning that uses a Bayesian representation of the uncertainty over models. The approach, BOSS (Best of Sampled Set) [4], drives exploration by sampling multiple models from the posterior, and plans by selecting actions optimistically. It extends previous work by providing a rule for deciding when to resample and how to combine the models.

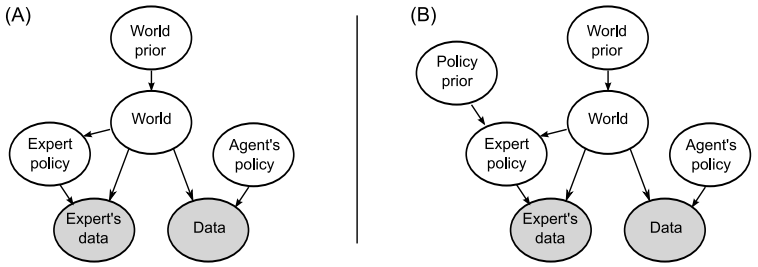

Figure 9: Two model of the generation of expert data. On the left: a model where we assume we have access to the same planning algorithm used by the expert. On the right: a model where we either do not have access to a good planner, or that the expert is only planning approximately.

We also provide formal guarantees, using the PAC framework, about our method's performance: we show that our algorithm achieves near-optimal reward with high probability with a sample complexity that is low relative to the speed at which the posterior distribution converges during learning. Empirically, we demonstrate that BOSS performs quite favorably compared to state-of-the-art reinforcement-learning approaches and illustrate its flexibility by pairing it with a non-parametric model that generalizes across states.

Unlike the ILEM model, which learns factorized state spaces, the BOSS model is a nonparametric state clustering model: states are clustered based on observed attributes, and the clusters are used to share statistical strength and improve estimates of transition probabilities. For example, the model is capable of understanding that states with one key feature might all have one kind of dynamic, while states with another key feature have a different kind of dynamic. Experience in one state can therefore be generalized to other states, resulting in radical generalization and improved transition estimate. These improved estimates result in improved control performance.

Fig. 8 provides some representative empirical results. An agent must navigate a maze (left); the right shows performance (vertical axis) as a function of time (horizontal axis) for different algorithms. BOSS outperforms other methods because of its ability to quickly cluster states together, even in the presence of uncertainty.

There are many unanswered questions about the BOSS model. While optimistic mixing of models is practically simple, it does not scale well with the size of the action space, nor the number of samples used. In addition, it assumes that exact planning is performed on the sampled world models; a natural extension is to consider approximate planning algorithms.

5 Planning and Model Building with Expert Information

Both the ILEM and BOSS build models based only on an agent's own interactions with an unknown world. How can expert demonstrations of (near) optimal behavior improve learning and control? We have also considered learning in partially observable domains where the agent can query an expert for just such near-optimal trajectories. This resulted in a nonparametric Bayesian

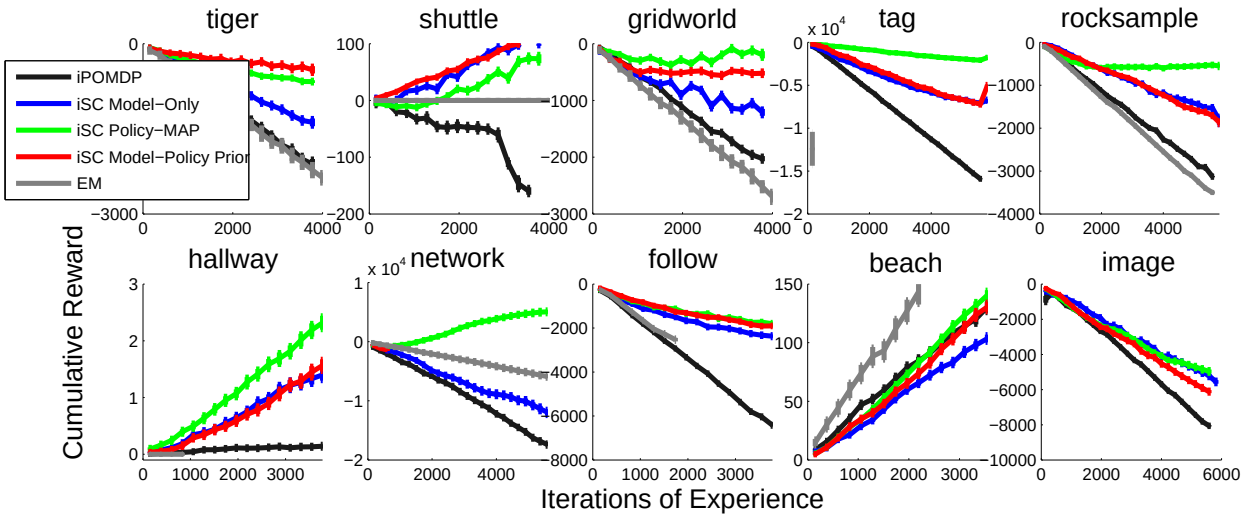

Figure 10: Learning curves on several standard POMDP problems. Error bars are 95% confidence intervals of the mean.

approach that combines model knowledge, which can be learned both from expert information and independent exploration, with policy knowledge, which can be inferred from expert trajectories [2]. Our approach additionally biases the agent towards models which are simple and controllable, which ultimately results in improved policy and model learning.

Combining data from independent exploration and expert trajectories is challenging: data gathered from independent observation provides information directly about the model dynamics and immediate rewards, whereas the expert demonstrations, by showing outputs of good policies, provide only indirect information about the underlying model. Similarly, immediate rewards observed during independent exploration provide indirect information about good policies.

Fig. 9 illustrates our approach. We use a Bayesian model-based RL approach to take advantage of both forms of data, applying Bayes rule to write a posterior over models M given data D as $p(M|D) \propto p(D|M)p(M)$. Different forms of this prior $p(M)$ lead us to three different learning algorithms: (1) if we know the expert's planning algorithm, we can sample models from $p(M|D)$, invoke the planner, and weigh models given how likely it is the planner's policy generated the expert's data; (2) if, instead of a planning algorithm, we have a *policy prior*, we can similarly weight world models according how likely it is that probable policies produced the expert's data; and (3) we can search directly in the policy space guided by probable models.

Fig. 10 illustrate results on benchmark POMDP tasks. The results show that the addition of expert data always either improves performance or makes no difference. Additional results (found in the paper) show that our extra bias towards controllability improves performance even without expert data.

The idea of policy priors represents a novel contribution to the field of Bayesian RL. While many researchers have placed priors over world models, far fewer have considered what policy priors

are, and how they can be used. We have used them in one particular context, but it is likely that others are possible. Leveraging or learning general procedural knowledge, as opposed to declarative knowledge, is one possibility; a more direct application may be controller synthesis or motor primitive learning. Exploring other uses – especially in cases where exact planning is not possible – remains an important open problem.

6 Hierarchical Planning

In complicated planning problems – such as those reminiscent of the "real world," – planning must occur at multiple levels of abstraction. This includes "task level" planning (for example, a cleaning agent may decide to dust before it vaccuums) and the motion planning level (what control signals are needed to actually move the vaccuum around?). We have developed a novel approach [8] to the integration of task planning and motion planning that has the following two key properties:

- It is aggressively hierarchical. It makes choices and commits to them in a top-down fashion in an attempt to limit the length of plans that need to be constructed, and thereby exponentially decrease the amount of search required. Importantly, our approach also limits the need to project the effect of actions into the far future.
- It uses goal regression, constructing partial symbolic descriptions of desired subgoals and making queries in a continuous geometric representation of the initial state. It does not require a complete symbolic representation of the input geometry or of the geometric effect of the task-level operations.

One key challenge is connecting high-level symbolic descriptors with low-level geometric planning. We handle this by using geometric suggesters, which are fast, approximate geometric computations that help the high-level processes make appropriate choices. For example, it is possible to determine which objects need to be moved out of the way by planning a path for a conservatively grown object in the 3D workspace rather than in the high-dimensional configuration space of the robot.

We demonstrate the method on a complicated "cleaning task," which involves moving objects around in a systematic fashion. A flat symbolic planner would have required significant search to find the plan; a geometric planner in the full configuration space could never have started. Here, we solved it by solving 8 small planning problems, the biggest of which required a two-step plan, and also solved many simple motion plans for suggestions. Finally, we solved detailed robot-motion planning problems for each primitive action separately. The web site

http://people.csail.mit.edu/tlp/hierarchicalVideos/

contains movies of the robot solving the swap and wash examples, as well as several more complex problems. In all of these cases, we find a considerable decrease in planning horizon, which comes with an exponential decrease in the size of the space to be searched.

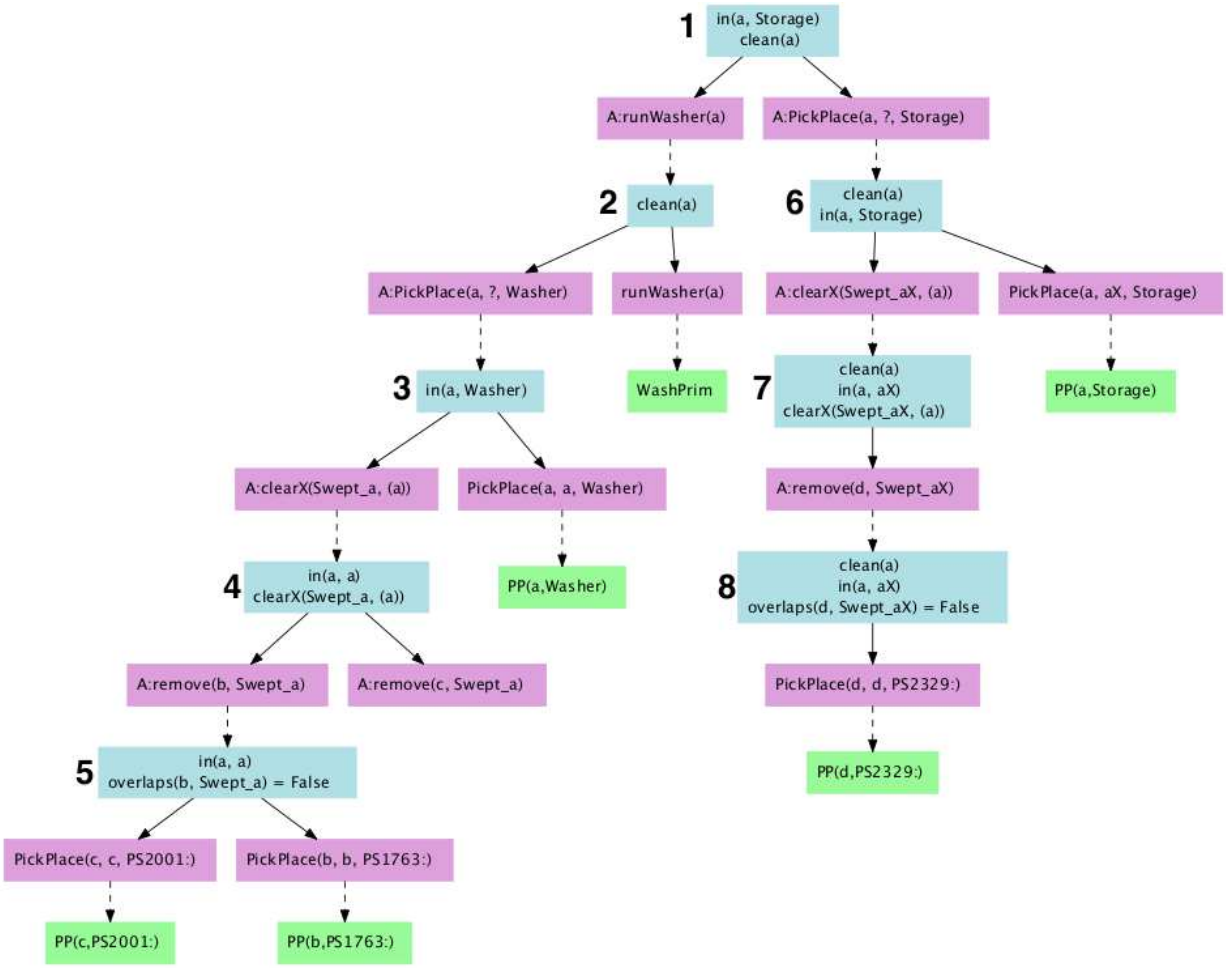

Figure 11: Planning and execution tree for washing and putting away an object. Dashed arrows are subtask refinements.

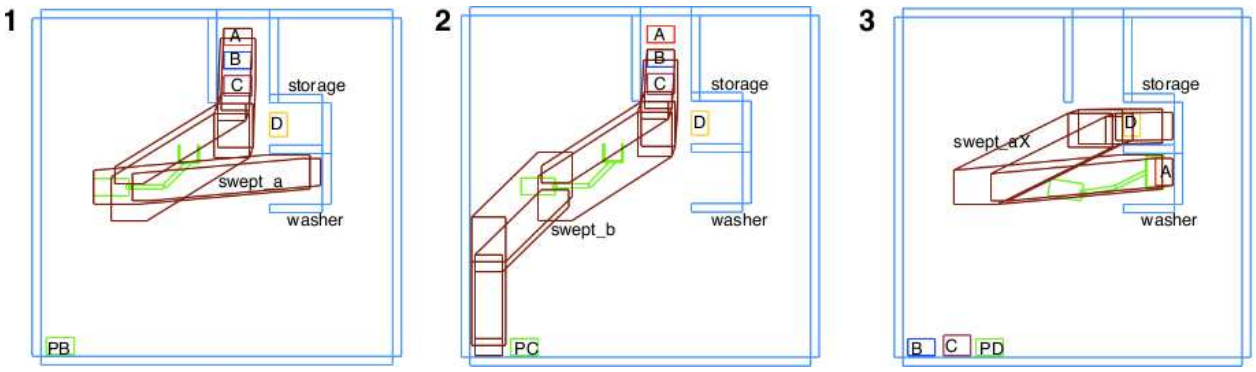

Figure 12: Suggestions for swept paths and parking locations.

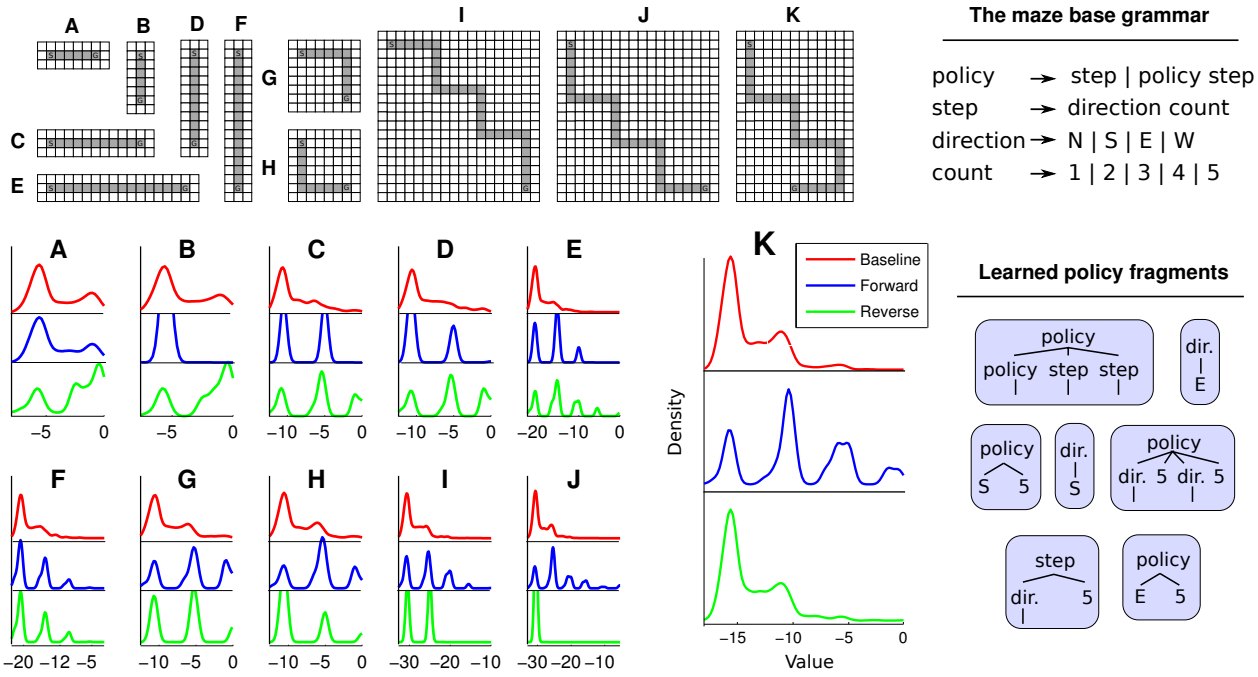

Figure 13: Transfer learning results on a sequence of increasingly complex mazes.

7 Transfer Learning with Hierarchical Bayesian Models

In order for decision making to be robust, an agent must be able to transfer knowledge gained in one situation to problems faced in another. Here, we discuss two different projects that use hierarchical Bayesian models to do transfer.

7.1 Compositional Transfer via Nonparametric Grammars

We first investigated a probabilistic framework for incorporating structured inductive biases into reinforcement learning [9]. These inductive biases arise from the policy priors previously discussed (ie, probability distributions over optimal policies). Borrowing recent ideas from computational linguistics and Bayesian nonparametrics, we define several families of policy priors that express compositional, abstract structure in a domain. Compositionality is expressed using probabilistic context-free grammars, enabling a compact representation of hierarchically organized sub-tasks. Useful sequences of sub-tasks can be cached and reused by extending the grammars nonparametrically using Fragment Grammars. We also developed Monte Carlo methods for performing inference, and show how structured policy priors lead to substantially faster learning in complex domains compared to methods without inductive biases.

Empirically, we tested our method on a family of increasingly complex maze tasks, shown in Figure 13. We showed that fragment grammars are capable of capturing abstract, reusable bits of policy knowledge; this provides improved inductive biases upon entering new tasks. Importantly,

this knowledge exists at several levels of abstraction, from low-level information (such as specific motor primitives) to high-level patterns in the mazes (for example, the fact that most useful primitives occur in sequences of 5 primitive actions).

It is important to emphasize that this is a modeling *framework*, rather than a particular model: Each policy prior makes structural assumptions that may be suitable for some domains but not others. We described several families of policy priors, the most sophisticated of which (fragment grammars) can capture hierarchically organized, reusable policy fragments. Since optimal policies in many domains appear amenable to these structural assumptions, we believe that fragment grammars may provide a powerful inductive bias for learning in such domains.

7.2 Models of Human Modeling and Decision Making

Finally, we have made significant progres on using hierarchical Dirichlet processes to transfer knowledge across worlds in reinforcement-learning domains. We have shown how knowledge learned in one domain can be leveraged in another, and how transfer at different levels of abstraction can be accomplished by learning about different parameter settings in a unified model. The knowledge transferred can result both in fast learning of good models and in fast learning of optimal policies, depending on the overlap in structure between the previous worlds and the target world.

Our specific technical contribution is a Hierarchical Dirichlet process model. By performing joint inference over training and test data, our model is able to transfer knowledge at various levels of abstraction. This results in useful inductive biases about a new world after experiencing training worlds with varying amounts of consistency. Importantly, our model allows abstract regularities about the world to be transferred, such as the determinism in the world, or the aggressiveness with which states should be clustered – we illustrate how agents can learn both to be aggressive and conservative in their learning. This leads in turn to improved performance vis-a-vis cumulative reward when compared to other agents.

A few representative empirical results are shown in Fig. 14. In the "Stripe World," performance of our algorithm is measured (vertical axis) as a function of time (horizontal axis) for different amounts of overlap between training worlds and a test world. For the "Corridor World," performance is of our transfer method is compared to other algorithms.

Hierarchical Bayes provides a compositional, modular framework in which to think about transfer at multiple levels, both in terms of specific and abstract knowledge. Future work could extend our model by using other nonparametric distributions to capture other kinds of inductive bias, or more structured distributions over richer knowledge representations, to move towards more human-like generalization in RL.

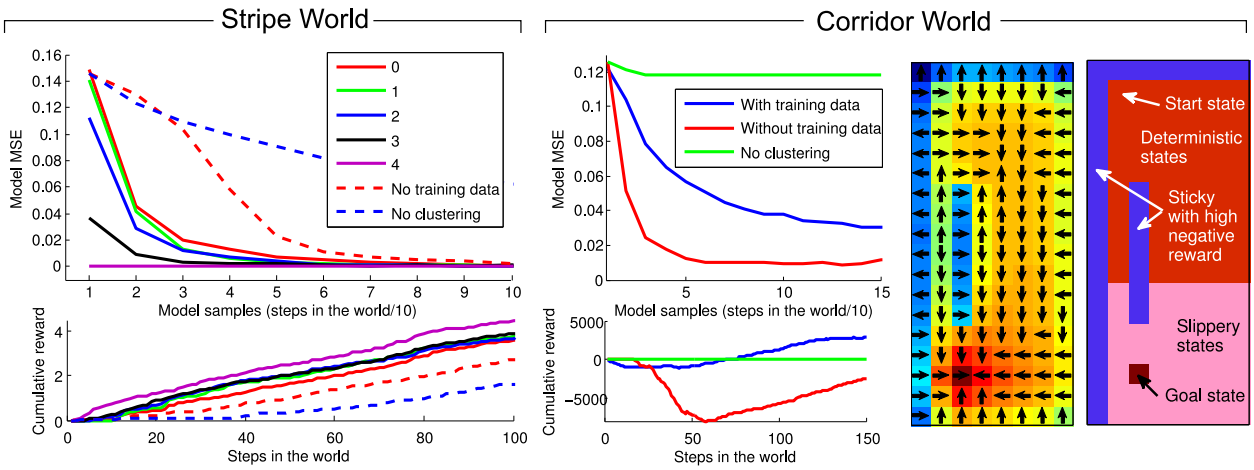

Figure 14: Transfer learning results. See text for details.

References

[1] Wingate, D., Stuhlmueller, A., Goodman, N.: Lightweight implementations of probabilistic programming languages via transformational compilation. In: Artificial Intelligence and Statistics (AISTATS). (2011)

[2] Wingate, D., Goodman, N., Stuhlmueller, A., Siskind, J.: Nonstandard interpretations of probabilistic programs for efficient inference. In: Neural Information Processing Systems (NIPS). (2011)

[3] Wingate, D., Weber, T., Kane, J.: A reinforcement learning approach to variational inference in probabilistic programming. In: (in prep). (2011)

[4] Asmuth, J., Li, L., Littman, M.L., Nouri, A., Wingate, D.: A Bayesian sampling approach to exploration in reinforcement learning. In: 25th Conference on Uncertainty in Artificial Intelligence (UAI'09). (2009)

[5] Wingate, D., Goodman, N.D., Roy, D.M., Tenenbaum, J.B.: The infinite latent events model. In: Uncertainty in Artificial Intelligence (UAI). (2009)

[6] Doshi-Velez, F., Wingate, D., Roy, N., Tenenbaum, J.: Infinite dynamic bayesian networks. In: International Conference on Machine Learning(ICML). (2011)

[7] Wingate, D., Goodman, N., Roy, D., Kaelbling, L., Tenenbaum, J.: Bayesian policy search with policy priors. In: International Joint Conference on Artificial Intelligence (IJCAI). Best poster award. (2011)

[8] Kaelbling, L.P., Lozano-Perez, T.: Hierarchical planning in the now. In: IEEE Conference on Robotics and Automation Workshop on Mobile Manipulation. (2010)

[9] Wingate, D., Gershman, S., Diuk, C., O'Donnell, T., Tenenbaum, J.: A probabilistic foundation for compositional policy priors. In: (in prep). (2011)